AF
SPW

P/1318-136

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:

Robert Barritz — Date: May 4, 2006

Serial No.: 09/933,540 — Group Art Unit: 2194

Filed: August 20, 2001 — Examiner: Van H. Nguyen

For: METHOD AND SYSTEM FOR DETERMINING THE USE AND NON-USE OF SOFTWARE PROGRAMS

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

SUPPLEMENTAL AMENDED APPEAL BRIEF PURSUANT TO 37 C.F.R. §1.192 IN RESPONSE TO THE NOTIFICATION OF NON-COMPLIANT APPEAL BRIEF

Sir:

This appeal is taken from the final action mailed August 10, 2005. In support of the Notice of Appeal filed November 4, 2005, this amended Appeal Brief is submitted in response to the Notification of Non-Compliant Appeal Brief mailed on March 22, 2006. It is respectfully submitted that this amended Appeal Brief complies with all of the requirements of 37 C.F.R. §41.37(c).

I. REAL PARTY IN INTEREST:

The real party in interest in the above-identified application is: Isogon Corporation.

II. RELATED APPEALS AND INTERFERENCES:

There are no related appeals or interferences of which applicant is aware regarding the above-identified application.

III. STATUS OF CLAIMS:

Claims 1, 9, 11-12, and 14-25 stand rejected under 35 U.S.C. §102(e).

Claims 2-7, 10, 13, 26-27, and 28-33 stand rejected under 35 U.S.C. §103(a)

Claim 8 has been canceled.

IV. STATUS OF AMENDMENTS:

A response to the final Office Action was filed on August 24, 2005. A Notice of Appeal was filed on November 4, 2005. An Advisory Action was mailed on September 20, 2005.

V. SUMMARY OF CLAIMED SUBJECT MATTER:

The invention is directed to an improved software auditing system and method whereby the execution of software modules can be determined in a manner that does not overly burden system resources. Execution information is filtered out and remaining information is correlated with other information to determine products which have been executed. The invention is concerned with the system resources, and is designed to obtain data for fewer than all of the load modules that execute on the system.

Referring now to independent claim 1, which is included in the focus of this appeal, a plurality of features are defined. In particular, claim 1 includes an operating system 24 of a computer 10 that controls execution in the computer of the software products through the invocation of respective load modules thereof. Further, claim 1 includes a monitor 22 that is periodically triggered to collect load module execution information (see page 6, lines 10-29).

Also, a filtering facility is provided that is effective to filter at least one previously identified program from the load module execution information (see page 13, lines 8-16, also page 15, lines 1-4). A correlator is further defined that correlates the filtered load module execution information with data that associates load module names with corresponding software products and develops a list of software products executed in the computer over the course of a given time period (see, for example, page 11, lines 5-17). A reporter 60 outputs data reflecting the use of the software products in the computer in terms of software product names thereof.

Referring now to claim 15, a system for determining program usage on a computer 10 is disclosed. The claim includes a plurality of executable software programs constituting software

products, each software product being constituted in turn of one or more load modules, the load modules being stored in at least one memory of the computer 10 (see Figure 5, page 22, line 6 – page 23, line 2). Further, an operating system 24 of the computer 10 is defined that controls execution in the computer of the software products through the invocation of respective load modules. A monitor 22 collects load module execution information by deducing which load modules are being used in given processes of the computer 10, without directly monitoring the actual invocation by the operating system of the load modules (page 11, lines 5-17).

Also, a filtering facility is provided to filter at least one previously identified program from the load module execution information (page 13, lines 8-16, also page 15, lines 1-4). A correlator correlates the filtered load module execution information with data that associates load module names with corresponding software products and develops a list of products executed in the computer over the course of a given time period (see, for example, page 11, lines 5-17). Also, a reporter 60 outputs data reflecting the use of the software products in the computer in terms of software product names thereof.

Referring now to claim 22, a system for determining program usage on a computer 10 is defined. The system in claim 22 comprises a plurality of executable software programs constituting software products, each of the software products being constituted of one or more load modules, the load modules being stored in at least one memory of the computer 10 (see Figure 5, page 22, line 6 – page 23, line 2). An operating system 24 of the computer 10 is defined that controls execution in the computer of software products through the invocation of respective load modules thereof. A monitor 22 collects software product execution information by monitoring input or output to specific files or datasets by the software products (page 11, lines 5-17). A filtering facility is defined that is effective to filter at least one previously identified program from the software product execution information, wherein such inputs and outputs are associated with corresponding software products or groups of software products (see page 10, line 23-page 11, line 4, also page 15, lines 1-4).

Referring now to claim 26, a system for determining non-usage of software products on a computer 10 is defined The system comprises a plurality of executable software programs constituting software products, each of the software products being constituted of one or more load modules, the load modules being stored in at least one memory of the computer 10 (see

Figure 5, page 22, line 6 – page 23, line 2). An operating system 24 of the computer 10 is further defined that controls execution in the computer 10 of software products through the invocation of respective load modules. A software product surveyor 12 surveys the at least one memory of the computer 10 and produces an inventory of the names of the load modules, the surveyor 12 being operable with an associator that identifies and associates and records associations between product names and the load module inventory names (see page 5, line 25-page 6, line 9).

Further, a monitor 22 collects load module execution information over a given time period (page 11, lines 5-17). A filtering facility that is effective to filter at least one previously identified program from the load module execution information (see page 10, line 23-page 11, line 4, page 15, lines 1-4). Also, a correlator that correlates the filtered load module execution information with data that associates load module names with corresponding software products and develops a list of products executed in the computer over the course of a selected time period (see, for example, page 11, lines 5-17). Further, a comparing facility that compares information provided by the correlator to information provided directly or indirectly by the surveyor and which produces a list of unused software products (see, for example, page 17, lines 4-14). Also, a reporter 60 that outputs data reflecting the non-use of software products in the computer 10.

Referring now to claim 28, a system for determining program usage on a computer 10 is defined. The system comprises a plurality of executable software programs constituting software products, each of the software products being constituted of one or more load modules, the load modules being stored in at least one memory of the computer 10 (see Figure 5, page 22, line 6 – page 23, line 2). Further, an operating system 24 of the computer 10 that controls execution in the computer of software products through the invocation of respective load modules thereof. A monitor 22 collects load module execution information reflecting the usage of software products on the computer 10 (page 11, lines 5-17). Also, a filtering facility is provided and effective to filter at least one previously identified program from the load module execution information (see page 13, lines 8-16, also page 15, lines 1-4).

Moreover, a library source determination facility is defined that determines the load library from which each executed load module has been loaded (see, for example, page 17, lines 15-21). Further, a reporter 60 is included that outputs data showing respective directory paths for load modules that have been executed.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL:

The following grounds of rejection are presented for review:

A. Rejection under 35 U.S.C. §102(e)

Whether claims 1, 9, 11-12, and 14-25 stand rejected under 35 U.S.C. §102(e) as being anticipated by Lin et al. ("Lin," U.S. Patent No. 5,949,415).

B. Rejection under 35 U.S.C. §103(a)

Further, whether claims 2-7, 10, 13, and 26-27 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Lin in view of Johnson (U.S. Patent No. 6,788,980).

C. Rejection under 35 U.S.C. §103(a)

Further, whether claims 28-33 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Lin in view of Evans (U.S. Patent No. 6,430,708).

VII. ARGUMENT:

A. Rejection under 35 U.S.C. §102(e)

Claims 1, 9, 11-12, and 14-25 stand rejected under 35 U.S.C. §102(e) as being anticipated by Lin et al. ("Lin," U.S. Patent No. 5,949,415). Applicant respectfully traverses this rejection.

The reference relied upon in this rejection is Lin et al. ("Lin," U.S. Patent No. 5,949,415). Respectfully, Lin, neither anticipates nor suggests the invention of claim 1. With respect to portions of claim 1, cited below, the Examiner has referred the applicant to column 9, lines 49-53 (claim 8) and column 10, lines 11-15 (claim 13) of Lin for support. Respectfully, careful study of these portions of Lin that are relied upon shows that this reference contains numerous differences and is structurally and functionally, as well, different from the claimed invention.

Contrary to the Examiner's characterization, Lin does not teach or suggest applicant's claimed monitor, filtering facility and correlator. Instead, Lin teaches a system task list that is already maintained by a computer's operating system and referenced to identify active tasks and subtasks. Applicant's claim 1, in contrast, defines that the monitor (not the operating system) collects load module execution information (see page 6, lines 10-29 of applicant's written

description).

Furthermore, Lin's detecting step merely extracts an identifier from the system task list already generated by the operating system. Applicant's claim 1 filtering facility, in contrast, filters at least one previously identified program from the load module execution information collected by the monitor. Lin's feature of merely extracting an identifier from an active task list is not tantamount to applicant's filtering facility that removes information from collected load module execution information. Lin teaches a well-known prior art step of extraction, which, unlike applicant's claim 1 system, does not conserve valuable system resources because an entire operating system task list (as opposed to a shorter filtered list) must be examined in order to extract a single identifier. Moreover, Lin does not teach or suggest applicant's claim 1 "correlator that correlates the filtered load module execution information with data that associates load module names with corresponding software products and develops a list of software products executed in the computer over the course of a given time period."

Thus, applicant's claim 1 features are taught or suggested by Lin. Respectfully, no description or structure is set forth anywhere in Lin's specification that would teach or suggest applicant's claim 1 "monitor," "filtering facility" or "correlator" to one skilled in the art. Therefore, applicant's claim 1 is patentable over Lin.

Applicant's independent claims 15 and 22 include similar limitations and, therefore, are also not anticipated by Lin. In particular, claim 15 defines a monitor that collects load module execution information "by deducing" and "without directly monitoring the actual invocation by the operating system of the load modules." Claim 15 further defines a filtering facility "that filters at least one previously identified program from the module load execution information." Lin does not teach or suggest applicant's claim 15 monitor or filtering facility.

Further, claim 22 includes a monitor that collects software product execution information by monitoring input or output to specific files. Claim 22 further defines a filtering facility, effective to filter at least one previously identified program from the software product execution information. As noted above with respect to claim 1, Lin neither teaches nor suggests applicant's claim 22 "monitor" and "filtering facility." Claims 9, 11-12, 14 and 16-25 depend directly or indirectly from claims 1, 15 or 22, respectively, and are, therefore, patentable for the same

reasons as well because of the features in those claims with the features set forth in the claim(s) from which they depend.

B. Rejection under 35 U.S.C. §103(a)

Claims 2-7, 10, 13, and 26-27 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Lin in view of Johnson (U.S. Patent No. 6,788,980). Applicant respectfully traverses this rejection.

Applicant notes that independent claim 26 also similar features as in the above-described claim 1 limitations. In particular, claim 26 includes a "monitor that collects load module execution information over a given time period" and a "filtering facility" operable to "filter at least one previously identified program from the load module execution information." Respectfully, Johnson does not teach or suggest applicant's claim 26 monitor and filtering facility that are missing from the teachings of Lin. Therefore, for at least the reasons set forth above, claim 26 is patentable combination of Lin and Johnson. Moreover, claims 2-7, 10 and 13 (which depend directly or indirectly from claim 1), and claim 27 (which depends from claim 26) are patentable for the same reasons, as well as because of the combination of the features in those claims with the features set forth in the claim(s) from which they depend.

C. Rejection under 35 U.S.C. §103(a)

Claims 28-33 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Lin in view of Evans (U.S. Patent No. 6,430,708). Applicant respectfully traverses this rejection.

Applicant notes that independent claim 28 also include the above-described claim 1 limitations, particularly the "monitor" and "filtering facility." Respectfully, Evans does not teach or suggest applicant's claim 28 "monitor that collects load module execution information reflecting the usage of software products on the computer," nor applicant's claim 28 "filtering facility that is effective to filter at least one previously identified program from the load module execution information" that are missing from the teachings of Lin. Therefore, for at least the reasons set forth above, claim 28 is patentable over the combination of Lin and Evans. Claims 29-33 (which depend from directly or indirectly from claim 28) are, therefore, patentable for the same reasons as well because of the features in those claims with the features set forth in the

claim(s) from which they depend.

VIII. CONCLUSION:

Check No. 23117 in the amount of $250.00 (small entity) to cover the fee for filing an Appeal Brief was previously submitted on January 6, 2006.

Applicant believes that no extension fee is due. However, if it is determined that this communication is filed after the shortened statutory time period had elapsed and no separate Petition is enclosed, the Commissioner of Patents and Trademarks is petitioned, under 37 C.F.R. §1.136(a), to extend the time for filing a response by the number of months which will avoid abandonment under 37 C.F.R. §1.135. The fee under 37 C.F.R. § 1.17 should be charged to our Deposit Account No. 15-0700.

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as First Class Mail in an envelope addressed to: Mail Stop Appeal Brief – Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450, on May 4, 2006:

Max Moskowitz
Name of applicant, assignee or Registered Representative

Signature

May 4, 2006
Date of Signature

Respectfully submitted,

Max Moskowitz
Registration No.: 30,576
OSTROLENK, FABER, GERB & SOFFEN, LLP
1180 Avenue of the Americas
New York, New York 10036-8403
Telephone: (212) 382-0700

MM:JJF:ck

CLAIMS APPENDIX

1. A system for determining program usage on a computer, the system comprising:

a plurality of executable software programs constituting software products, each of the software products being constituted of one or more load modules, the load modules being stored in at least one memory of the computer;

an operating system of the computer that controls execution in the computer of the software products through the invocation of respective load modules thereof;

a monitor that is periodically triggered to collect load module execution information;

a filtering facility that is effective to filter at least one previously identified program from the load module execution information;

a correlator that correlates the filtered load module execution information with data that associates load module names with corresponding software products and develops a list of software products executed in the computer over the course of a given time period; and

a reporter that outputs data reflecting the use of the software products in the computer in terms of software product names thereof.

2. The system of claim 1, in which the monitor operates by taking periodic snapshots of the then current state of active processes in the computer.

3. The system of claim 2, in which the load module execution information includes one or more of the following data items: module names, user names, the time when processes were started, how much CPU time has been used, and a directory path name from which processes were installed.

4. The system of claim 2, including a facility that allows adjusting the period between snapshots in response to program usage activity levels.

5. The system of claim 2, in which the monitor produces a list of executing load modules and their respective directory path names.

6. The system of claim 2, including a facility that determines how many processes have begun and ended between snapshots.

7. The system of claim 2, including a facility that compares successive snapshots to determine which modules have executed and how many were missed.

Claim 8 (canceled).

9. The system of claim 1, including in the load module execution information module names and other process related information including directory, start time, and process ID.

10. The system of claim 2, in which processes are identified by PID (Process IDs) numbers and the monitor includes a facility that obtains a measure of missed processes by calculating M minus H minus E, where M represents a current PID, H represents the highest PID found in a prior snapshot and E represents a count of all processes that have begun since the prior snapshot and are still executing.

11. The system of claim 1, in which the correlator operates in conjunction with a knowledge base that associates load module names with software product names.

12. The system of claim 1, further including a surveying program that develops an inventory of substantially all software products on the computer and a facility which produces a list of non-used software products based on comparing the inventory of software products against the data outputted by the reporter which reflects the use of the software products in the computer.

13. The system of claim 11, in which the knowledge base is a database of records which also associates file names to software products that use them and additionally includes at least one of the following: flags indicating if a module is used uniquely or shared among vendor products; a number indicating file matches required for correlation with the product; file type; file size; file creation date; and embedded strings of text.

14. The system of claim 1, in which the correlator operates by correlating module usage data with an inventory of software products that itself has been obtained by correlating in a knowledge base load module names with software product names.

15. A system for determining program usage on a computer, the system comprising;

a plurality of executable software programs constituting software products, each software product being constituted in turn of one or more load modules, the load modules being stored in at least one memory of the computer;

an operating system of the computer that controls execution in the computer of the software products through the invocation of respective load modules thereof;

a monitor that collects load module execution information by deducing which load modules are being used in given processes of the computer, without directly monitoring the actual invocation by the operating system of the load modules;

a filtering facility that is effective to filter at least one previously identified program from the load module execution information;

a correlator that correlates the filtered load module execution information with data that associates load module names with corresponding software products and develops a list of products executed in the computer over the course of a given time period; and

a reporter that outputs data reflecting the use of the software products in the computer in terms of software product names thereof.

16. The system of claim 15, in which the monitor obtains the load module execution information from a load module table created by the operating system which makes entries in the load module tables as processes are executed and access requests for load modules are made.

17. The system of claim 15, in which the monitor is implemented to execute every time the end of a process is reached.

18. The system of claim 15, in which the monitor executes as an exit routine near the end of a process.

19. The system of claim 15, in which the monitor gathers and accumulates usage data across sub processes of a higher level process so that when the load module table is successively read, only those module entries not previously encountered in a prior sub process of the current high level process are accumulated and names of load modules already found in the table for the current high level process are ignored.

20. The system of claim 15, in which the correlator operates by identifying the names of all software products used by correlating module usage data by using a knowledge base that associates the names of load modules with software products they comprise.

21. The system of claim 15, in which the correlator operates by correlating module usage data with an inventory of software products that itself has been obtained by correlating in a knowledge base load module names with software product names.

22. A system for determining program usage on a computer, the system comprising:

a plurality of executable software programs constituting software products, each of the software products being constituted of one or more load modules, the load modules being stored in at least one memory of the computer;

an operating system of the computer that controls execution in the computer of software products through the invocation of respective load modules thereof;

a monitor that collects software product execution information by monitoring input or output to specific files or datasets by the software products; and

a filtering facility that is effective to filter at least one previously identified program from the software product execution information, wherein such inputs and outputs are associated with corresponding software products or groups of software products.

23. The system of claim 22, in which the monitor is implemented as a background process.

24. The system of claim 22, in which the monitor is implemented as an intercept systematically placed in either or both of a file open and/or file close system function of the computer.

25. The system of claim 22, in which the monitor is operated as a batch process.

26. A system for determining non-usage of software products on a computer, the system comprising:

a plurality of executable software programs constituting software products, each of the software products being constituted of one or more load modules, the load modules being stored in at least one memory of the computer;

an operating system of the computer that controls execution in the computer of software products through the invocation of respective load modules thereof;

a software product surveyor that surveys the at least one memory of the computer and produces an inventory of the names of the load modules, the surveyor being operable with an associator that identifies and associates and records associations between product names and the load module inventory names;

a monitor that collects load module execution information over a given time period;

a filtering facility that is effective to filter at least one previously identified program from the load module execution information;

a correlator that correlates the filtered load module execution information with data that associates load module names with corresponding software products and develops a list of products executed in the computer over the course of a selected time period;

a comparing facility that compares information provided by the correlator to information provided directly or indirectly by the surveyor and which produces a list of unused software products; and

a reporter that outputs data reflecting the non-use of software products in the computer.

27. The system of claim 26, in which the comparator is constructed to delete from the inventory those software product names which have been detected by the monitor as having been

executed in the computer at least once during the given time period, leaving a list of non-used software products.

28. A system for determining program usage on a computer, the system comprising:

a plurality of executable software programs constituting software products, each of the software products being constituted of one or more load modules, the load modules being stored in at least one memory of the computer;

an operating system of the computer that controls execution in the computer of software products through the invocation of respective load modules thereof;

a monitor that collects load module execution information reflecting the usage of software products on the computer;

a filtering facility that is effective to filter at least one previously identified program from the load module execution information;

a library source determination facility that determines the load library from which each executed load module has been loaded; and

a reporter that outputs data showing respective directory paths for load modules that have been executed.

29. The system of claim 28, in which the library source determination facility obtains a list of modules that have been used by a particular process, determines the load libraries and their search order used by the process, and using a search order determined in a prior step, searches the load libraries of the computer for a first library containing the same modules that best matches the list of modules used.

30. The system of claim 28, in which the monitor operates as a concurrent process and module usage data and load library collection data are both obtained by the monitor and library usage is concurrently determined.

31. The system of claim 29, in which the task of determining the correct load libraries in their appropriate search order is carried out as a separate process, wherein one of the module

usage data or library collection data is obtained by the monitor and the other is obtained from a separate source and processed to determine load library usage.

32. The system of claim 28, in which the library source determination facility uses a JCL (Job Control Language) interpreter.

33. The system of claim 28, in which the load library determination facility determines both the identity and order of load libraries used by a particular process by reading job control language ("JCL") data structures of a current job to obtain a load library list for the process.

EVIDENCE APPENDIX

None.

RELATED PROCEEDINGS APPENDIX

None.